“Modalin” Database Development

One of Indonesia’s Largest Used Car Communities

MODALIN

Your used-car trusted partner

Outline

1. Introduction
    a. Background
    b. Objective
2. Database Planning
    a. Entity & Attribute Identification
    b. Relation and Entity Identification
    c. Defining Business Rule
    d. ERD Design
3. Database Implementation
    a. Develop the Database
    b. Develop Table on Database
    c. Add Constraint from Business Rule on Table
4. Populating Database
    a. Develop Dummy Data
    b. Import Dummy Data into table
    c. Database Backup
5. Transactional and Analytical Query
    a. Transactional Query - Case Explanation
    b. Analytical Query - Case Explanation

1. Introduction

a. Background

Company Overview

Modalin is a C2C (Customer to Customer) startup based in Pamekasan, East Java. The company was founded with the mission of creating a user-friendly and convenient platform for the exchange of used vehicles.

Main Product

Modalin's main offering is a Facebook-based community that serves various types of automotive enthusiasts, including those interested in used car trading. Through the community group, users can connect with other users and trade or exchange used vehicles with ease. The platform provides a range of tools and resources to listing service, bidding, and messaging system. For the transaction process, currently is occurred on out of website.

Vision

Modalin’s goal is to provide a **user-friendly** and **convenient experience** for all parties, ultimately facilitating the exchange of used vehicles in a seamless and hassle-free manner.

b. Objective

To design and develop a relational database for a used car sales website that can support listing and bidding features, as well as user profile management, efficient search functionality, and other necessary database operations, to ensure a smooth and user-friendly experience for both buyers and sellers.

2. Database Planning

a. Requirement

Several critical activities that need to be accommodated on the website:

i. Upload listed car into web
ii. Advertise uploaded vehicle into web
iii. Bid for interested vehicle

b. Entity and Attribute Identification

In this section will be explained related to entity and attribute required for each entity.

Entity	Definition	Attribute
Users	This entity represents the users of the website who can offer cars for sale or search for cars to buy.	• user_id, • name, • contact
Car	This entity represents the cars that are being offered for sale.	• product_id, • brand, • model, • body_type, • year, • price, • user_id, *– referenced to user that upload the car* • kota_id *– referenced to location where the car is offered*
City	This attribute represents the city where the car offered is located.	• kota_id, • nama_kota, • latitude, • longitude
Advertisement	This entity represents the advertisements for the cars that are being offered for sale	• ad_id, • date_post, • status, • product_id, *– referenced to car that offered by user (seller)* • user_id *– referenced to user (seller)*
Bid	This attribute represents the car that customer (buyer) that want to bid	• bid_id, • bid_price, • bid_status, • bid_date, • ad_id, *– referenced to available advertisement* • buyer_id *– referenced to user (buyer)*

For the SQL Query related to scheme development of the entities above is attached on Appendix

c. Relation Identification

Relationship Requirement:

1. A user (seller) is able to have more than 1 offered car
2. A car is only able to be owned by 1 user (seller)
3. An advertisement is only related to 1 car, 1 vehicle, and 1 location (city)
4. A bid is only related to 1 user (buyer) and 1 advertising

Relationship Table

Relationship	Users	Car	City	Advertisement	Bid
Users		1:N			1:N
Car				1:1	
City		1:N			
Advertisement					1:N
Bid					

Remark:

1:N - One-to-Many Relationship
1:1 - One-to-One Relationship

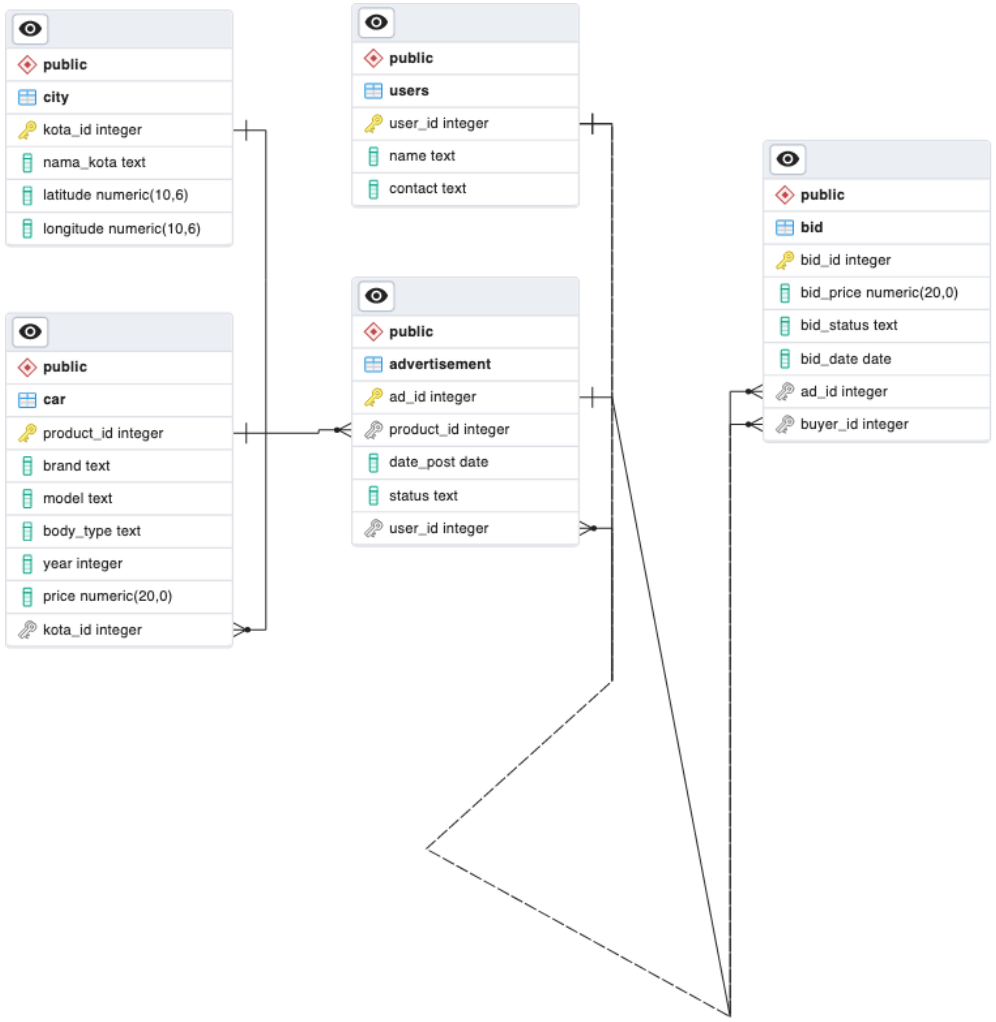
d. Defining Business Rule

The business rules for this project are:

Table	**Business Rule**
Users	● Each user must have a valid name ● Each user must have a valid contact information
Car	● The car price (price) must always be positive and not zero ● Each car must have a valid brand and body type
City	● Each city must have a latitude and longitude ● Each city name is can't be NULL
Advertisement	● Each advertisement must have a valid posting date (date_post) ● Each advertisement must have either an 'Available' or a 'Sold' status
Bid	● Each bid must have a valid bid date (bid_date). ● Each bid value must always be positive and not zero ● A user can only make one bid on each advertisement. ● A user isn't able to bid on their own offered vehicle

e. ERD Design

This sub-section will be explained related to ERD (Entity Relationship Diagram) of the database:



3. Database Implementation

a. Develop the Database

This sub-section will be explained related to database development with the name ‘Modalin’

```
CREATE DATABASE modalin
    WITH
    OWNER = postgres
    ENCODING = 'UTF8'
    CONNECTION LIMIT = -1
    IS_TEMPLATE = False;
```

b. Develop Table on Database

This sub-section will be explained related to table development on the database built

Scheme	SQL - Table Development
Users	CREATE TABLE Users (user_id SERIAL PRIMARY KEY, name TEXT NOT NULL, contact TEXT NOT NULL);
City	CREATE TABLE City (kota_id SERIAL PRIMARY KEY, nama_kota TEXT NOT NULL, latitude NUMERIC(10,6) NOT NULL, longitude NUMERIC(10,6) NOT NULL);
Car	CREATE TABLE Car (product_id SERIAL PRIMARY KEY, brand TEXT NOT NULL, model TEXT NOT NULL, body_type TEXT NOT NULL, year INT NOT NULL, price NUMERIC(10,2) NOT NULL, kota_id INT NOT NULL REFERENCES City(kota_id),);
Advertisement	CREATE TABLE Advertisement (ad_id SERIAL PRIMARY KEY, product_id INT UNIQUE REFERENCES Car(product_id), date_post DATE NOT NULL, status TEXT NOT NULL, user_id INT NOT NULL REFERENCES Users(user_id));
Bid	CREATE TABLE Bid (bid_id SERIAL PRIMARY KEY, bid_price NUMERIC(10,2) NOT NULL, bid_status TEXT NOT NULL, bid_date DATE NOT NULL, ad_id INT NOT NULL REFERENCES Advertisement(ad_id), buyer_id INT NOT NULL REFERENCES Users(user_id));

c. Add Constraint from Business Rule on the Table

From the business rules above, there are the constraints that we need to enable in order to make the scheme run as expected:

Scheme	Constraint
Users	• Each user must have a valid name *-- has been added directly on table creation* • Each user must have a valid contact information *-- has been added directly on table creation*
City	• Each city must have a latitude and longitude *-- have been added directly on table creation* • Each city name is can't be NULL *-- has been added directly on table creation*
Car	• The car price (price) must always be positive and not zero ALTER TABLE Car ADD CONSTRAINT CHK_Car_Price CHECK (price > 0); • Each car must have a valid brand and body type *-- have been added directly on table creation*
Advertisement	• Each advertisement must have a valid posting date (date_post) *-- has been added directly on table creation* • Each advertisement must have either an 'Available' or a 'Sold' status ALTER TABLE Advertisement ADD CONSTRAINT CHK_Advertisement_Status CHECK (status IN ('Available', 'Sold'));
Bid	• Each bid must have a valid bid date (bid_date). *-- have been added directly on table creation* • Each bid value must always be positive and not zero ALTER TABLE Bid ADD CONSTRAINT CHK_Bid_Price CHECK (bid_price > 0); • A user can only make one bid on each advertisement. ALTER TABLE Bid ADD CONSTRAINT UQ_Bid_User_Advertisement UNIQUE (buyer_id, ad_id); • A user isn't able to bid on their own offered vehicle CREATE FUNCTION check_bid_user(bid_row Bid) RETURNS boolean AS $$

Scheme	Constraint

	BEGIN
	RETURN NOT EXISTS (
	SELECT 1
	FROM Advertisement
	WHERE ad_id = bid_row.ad_id
	AND user_id = bid_row.buyer_id
	);
	END;
	$$ LANGUAGE plpgsql;
	
	ALTER TABLE Bid
	ADD CONSTRAINT CHK_Bid_BuyerSeller
	CHECK (check_bid_user(Bid) = true);

4. Populating Database

a. Develop Dummy Data

In this chapter will be explained related to table that modified or fully generated. The python code of the logic below is attached

Scheme	Python Logic
Users	Using Faker in develop random value for: • name • contact and also use the serial number to fill user_id
City	- No modification on this table, but the data is imported to generate value as a reference for kota_id in car table
Car	Modify given car table through add 2 more columns: user_id and kota_id • kota_id using random choice by city table
Advertisement	All of this column is created with logic as follow: • ad_id, product_id are using a serial value between 1 and 50 • date_post is using random integer between 2022-2023 • status is using random.choice between ‘Available’ and ‘Sold’ • user_id random_int from 1,51

Bid	All of this column is created with logic as follows: • bid_id is using serial value between 1 and 50 • bid_price is using random integer between 100 Mio and 250 Mio Rupiahs (with a multiple of 1000) • bid_status is using random status between 'Placed', 'Accepted', 'Rejected • bid_date is using random integer between 2022-2023 • ad_id, buyer_id are using random integer between 1 and 50

b. Import Dummy Data into table

After the data were ready, I upload the data into the SQL through convert data on the table into SQL Language using chatgpt.

During the upload process, there are some constraints that work well.

a. User identified to make more than 1 bid into same vehicle → we need to modify a bit then.

```
ERROR:  duplicate key value violates unique constraint "uq_bid_user_advertisement"
DETAIL:  Key (buyer_id, ad_id)=(49, 49) already exists.
SQL state: 23505
```

b. User identified make bidding into their own offered vehicle → we need to modify a bit then.

```
ERROR:  new row for relation "bid" violates check constraint "chk_bid_buyerseller"
DETAIL:  Failing row contains (44, 172095000, Accepted, 2022-03-04, 44, 50).
SQL state: 23514
```

c. Database Backup

After the dataset is fully uploaded - then we back up the database.

5. Transactional and Analytical Query

Transactional Query

a. Case 1 - Look for car with year >= 2015

Total found **40 data** that have year 2015 or later

product_id	brand	model	year	price
46	Suzuki	Suzuki Ertiga	2015	125000000
34	Honda	Honda Jazz	2015	192000000

product_id	brand	model	year	price
18	Daihatsu	Daihatsu Ayla	2015	96000000
39	Honda	Honda CR-V	2016	269000000
8	Toyota	Toyota Agya	2016	110000000
14	Toyota	Toyota Calya	2016	104000000
48	Suzuki	Suzuki Ertiga	2016	168000000
1	Toyota	Toyota Yaris	2016	175000000
15	Toyota	Toyota Calya	2016	107000000
16	Daihatsu	Daihatsu Ayla	2016	83000000
24	Daihatsu	Daihatsu Terios	2017	166000000
10	Toyota	Toyota Agya	2017	115500000
13	Toyota	Toyota Calya	2017	115500000
19	Daihatsu	Daihatsu Ayla	2017	115000000
20	Daihatsu	Daihatsu Ayla	2017	113000000
28	Daihatsu	Daihatsu Xenia	2017	135000000
37	Honda	Honda CR-V	2017	345000000
47	Suzuki	Suzuki Ertiga	2018	178000000
25	Daihatsu	Daihatsu Terios	2018	201000000
2	Toyota	Toyota Yaris	2018	215000000
30	Daihatsu	Daihatsu Xenia	2018	159000000
31	Honda	Honda Jazz	2018	236000000
33	Honda	Honda Jazz	2018	216000000
36	Honda	Honda CR-V	2018	415000000
50	Suzuki	Suzuki Ertiga	2018	167000000
40	Honda	Honda CR-V	2018	398500000
41	Honda	Honda Civic	2018	350000000
21	Daihatsu	Daihatsu Terios	2018	190000000
32	Honda	Honda Jazz	2019	250000000
26	Daihatsu	Daihatsu Xenia	2019	170000000
17	Daihatsu	Daihatsu Ayla	2019	120000000
6	Toyota	Toyota Agya	2019	114000000
12	Toyota	Toyota Calya	2019	137000000
23	Daihatsu	Daihatsu Terios	2019	189900000
11	Toyota	Toyota Calya	2019	130000000
45	Honda	Honda Civic	2019	397500000
4	Toyota	Toyota Yaris	2020	220000000
9	Toyota	Toyota Agya	2022	155500000
22	Daihatsu	Daihatsu Terios	2022	223000000
27	Daihatsu	Daihatsu Xenia	2022	220500000

b. Case 2 - Add 1 new row into Bid

```
INSERT 0 1

Query returned successfully in 47 msec.
```

c. Case 3 - Look all car that are offered by 1 account, ordered by the latest.

There are 2 steps that I did. The first one is I would like to know which user offer the most quantity of cars on the website.

	user_id integer	count_product bigint
1	22	5
2	35	4
3	36	2
4	30	2
5	21	2
6	28	2
7	6	2
8	24	2
9	49	2
10	1	2

After that, we find that a user with user_id = 22 is offered with the highest quantity (5) compared with other users. Then, we look the detail of this user offered.

product_id integer	brand text	model text	year integer	price numeric (20)	date_post date
50	Suzuki	Suzuki Ertiga	2018	167000000	2023-04-25
9	Toyota	Toyota Agya	2022	155500000	2023-01-25
47	Suzuki	Suzuki Ertiga	2018	178000000	2022-10-20
45	Honda	Honda Civic	2019	397500000	2022-05-23
2	Toyota	Toyota Yaris	2018	215000000	2022-01-13

d. Case 4 - Look all car that contained “Yaris” and ordered by the lowest price

product_id integer	brand text	model text	year integer	price numeric (20)
5	Toyota	Toyota Yaris	2012	124000000
3	Toyota	Toyota Yaris	2014	162000000
1	Toyota	Toyota Yaris	2016	175000000
2	Toyota	Toyota Yaris	2018	215000000
4	Toyota	Toyota Yaris	2020	220000000

e. Case 5 - Look nearest ad that is located in 1 city with kota_id = 3173 consider latitude and longitude

product_id [PK] integer	brand text	model text	year integer	price numeric (20)	distance numeric
19	Daihatsu	Daihatsu Ayla	2017	115000000	0.0
20	Daihatsu	Daihatsu Ayla	2017	113000000	0.0

Analytical Case

a. Case 1 - Popularity of each car that receives Bid

Car model popularity ranking based on bid frequency

model text	count_product bigint	count_bid bigint
Suzuki Ertiga	5	16
Honda Civic	5	13
Honda CR-V	5	5
Honda Jazz	5	5
Toyota Calya	5	3
Daihatsu Terios	5	3
Daihatsu Xenia	5	2
Toyota Yaris	5	1
Toyota Agya	5	1
Daihatsu Ayla	5	1

b. Case 2 - Compare car price with an average price per city

Sample data from 50 rows of data. From the data below we can check the comparison of price per car with average car price per city for Makassar and Surabaya

	nama_kota text	brand text	model text	year integer	price numeric (20)	avg_car_city numeric
1	Kota Makassar	Toyota	Toyota Agya	2016	110000000	115166666.7
2	Kota Makassar	Toyota	Toyota Agya	2017	115500000	115166666.7
3	Kota Makassar	Daihatsu	Daihatsu Ayla	2019	120000000	115166666.7
4	Kota Surabaya	Toyota	Toyota Yaris	2012	124000000	138250000.0
5	Kota Surabaya	Daihatsu	Daihatsu Xenia	2014	100000000	138250000.0
6	Kota Surabaya	Honda	Honda Jazz	2015	192000000	138250000.0
7	Kota Surabaya	Toyota	Toyota Calya	2019	137000000	138250000.0

c. Case 3 - Compare first bid and next bid in one certain user

I try to use model ‘Ertiga’ as sample, there is 1 user that did recursive bidding. Previously he bid for 143.1 Mio (IDR), and the next bid he offer for 221.4 Mio (IDR)

model text	user_id integer	bid_date date	next_bid_price date	bid_price numeric (20)	next_bid_price numeric
Suzuki Ertiga	30	2022-05-10	[null]	201318000	[null]
Suzuki Ertiga	22	2022-05-17	2023-12-19	143193000	221479000
Suzuki Ertiga	21	2022-10-01	[null]	193749000	[null]
Suzuki Ertiga	40	2023-06-08	[null]	242059000	[null]
Suzuki Ertiga	22	2023-12-19	[null]	221479000	[null]

d. Case 4 - Compare the percentage difference in the average car price by model and the average bid price offered by customers in the last 6 months

There are 10 car **model**, **avg_price** (consider all time), **avg_bid_6months** (bid_price that considered is bid_price that has date difference with today (‘2023-04-15) is less than or equal 180 days), **difference**, and **differenct_percent**

	model text	avg_price numeric	avg_bid_6month numeric	difference numeric	different_percent numeric
1	Daihatsu Ayla	105400000.00	0	105400000.00	100.00
2	Daihatsu Terios	193980000.00	159966500.00	34013500.00	17.53
3	Daihatsu Xenia	156900000.00	115319500.00	41580500.00	26.50
4	Honda CR-V	302083333.33	139142000.00	162941333.33	53.94
5	Honda Civic	266357142.86	191682375.00	74674767.86	28.04
6	Honda Jazz	210666666.67	183580333.33	27086333.34	12.86
7	Suzuki Ertiga	140687500.00	195961090.91	-55273590.91	-39.29
8	Toyota Agya	118400000.00	0	118400000.00	100.00
9	Toyota Calya	118700000.00	184557000.00	-65857000.00	-55.48
10	Toyota Yaris	179200000.00	199385000.00	-20185000.00	-11.26

e. Case 5 - Develop a window function average bid price of a car make and model over the last 6 months.

There are 6 columns additional from last 6 months until last 1 months bid.

brand text	model text	avg_bid_6month numeric	avg_bid_5month numeric	avg_bid_4month numeric	avg_bid_3month numeric	avg_bid_2month numeric	avg_bid_1month numeric
Daihatsu	Daihatsu Ayla	0	0	0	0	0	0
Daihatsu	Daihatsu Terios	159966500.00	159966500.00	159966500.00	159966500.00	105566000.00	105566000.00
Daihatsu	Daihatsu Xenia	115319500.00	115319500.00	115319500.00	115319500.00	115319500.00	115319500.00
Honda	Honda CR-V	139142000.00	139142000.00	139142000.00	139142000.00	139142000.00	139142000.00
Honda	Honda Civic	191682375.00	191682375.00	191682375.00	187884857.14	195869750.00	189973000.00
Honda	Honda Jazz	183580333.33	183580333.33	183580333.33	183580333.33	195102500.00	195102500.00
Suzuki	Suzuki Ertiga	195961090.91	202251500.00	209419000.00	203837714.29	203837714.29	203837714.29
Toyota	Toyota Agya	0	0	0	0	0	0
Toyota	Toyota Calya	184557000.00	184557000.00	184557000.00	135404000.00	135404000.00	135404000.00
Toyota	Toyota Yaris	199385000.00	199385000.00	199385000.00	199385000.00	199385000.00	199385000.00

End.